Exact Graph Structure Estimation with Degree Priors

Bert Huang and Tony Jebara
Computer Science Department
Columbia University
New York, New York 10027
{bert,jebara}@cs.columbia.edu

Abstract

We describe a generative model for graph edges under specific degree distributions which admits an exact and efficient inference method for recovering the most likely structure. This binary graph structure is obtained by reformulating the inference problem as a generalization of the polynomial time combinatorial optimization known as b-matching. Standard b-matching recovers a constant-degree constrained maximum weight subgraph from an original graph instead of a distribution over degrees. After this mapping, the most likely graph structure can be found in cubic time with respect to the number of nodes using max flow methods. Furthermore, in some instances, the combinatorial optimization problem can be solved exactly in near quadratic time by loopy belief propagation and max product updates even if the original input graph is dense. We show an example application to post-processing of recommender system predictions.

1 Introduction

An important task in graph analysis is estimating graph structure given only partial information about nodes and edges. This article will consider finding subgraphs from an original (possibly fully connected and dense) graph, subject to information about edges in terms of their weight as well as degree distribution information for each node.

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Such a graph contains an exponential number of subgraphs (graphs that can be obtained from the original by performing edge deletion). In fact, the number of subgraphs is $2^{|\mathcal{E}|}$, and since $|\mathcal{E}|$ can be up to $|\mathcal{V}|(|\mathcal{V}|-1)/2$, search or probabilistic inference in such a space may often be intractable. Working with a probability distribution over such a large set of possibilities is not only computationally difficult but may also be misleading since some graph structures are known to be unlikely *a priori*. This article proposes a particular distribution over graphs that uses factorization assumptions and incorporates prior distributions over node degrees. We perform *maximum a posteriori* (MAP) estimation under this distribution by converting the problem into a maximum weight b-matching.

This conversion method generalizes maximum weight b-matching, which is applied to various classical applications such as advertisement allocation in search engines [11], as well as machine learning applications such as semi-supervised learning [7], and embedding of data and graphs [17, 18]. Our method also generalizes bd-matching (which itself is a generalization of b-matching) and k-nearest neighbors.

Previous efforts that exploit degree distribution information to denoise edge observations have relied on *approximate* loopy belief propagation, which suffered from local minima [12]. This article indicates that, in some settings, MAP estimation over subgraphs with degree priors can be solved *exactly* in polynomial time. Given our proposed conversion method, which formulates the problem as a b-matching, we can efficiently solve for the the optimal graph structure estimate even with degree distribution information.

Applications of our proposed method include situations in which degree information is inferred from statistical sampling properties, from empirical methods in which degree distributions are learned from data, or from more classical problems in which the degree probabilities are given. An example of the latter case is in protein interaction prediction, where 3D shape analysis can bound the number of mutually accessible binding sites of a protein [8]. Similarly, in some social network applications, the number of connections for each user may be known even though the explicit identities of the users who are connected to them are hidden (e.g., LinkedIn.com).

1.1 Outline

The remainder of the paper is organized as follows. In Section 2, we derive the main algorithm for MAP graph estimation with degree priors, prove its correctness, and discuss its computational cost and the methods it generalizes. In Section 3, we demonstrate one application of the method to post-processing graph predictions. Finally, we conclude in Section 4 with a brief summary and discuss some possible alternative applications and future work.

2 MAP Edge Estimation

In this section, we provide the derivation and prove the correctness of a method for maximizing a probability function defined over subgraphs. Using this method, we find the optimum of a distribution defined by a concave potential function over node degrees in addition to the basic local edge potentials. If we consider the degree potentials prior probabilities over node degrees, the operation can be described as a *maximum a posteriori* optimization.

Formally, we are interested in finding a subgraph of an original graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. First, consider a distribution over all possible subgraphs that involves terms that factorize across (a) edges (to encode independent edge weight) and (b) degree distribution terms that tie edges together, producing dependencies between edges. We assume the probability of any candidate edge set $\hat{\mathcal{E}} \subseteq \mathcal{E}$ is expressed as

$$\Pr(\hat{\mathcal{E}}|\mathcal{G}) \propto \prod_{(i,j)\in\hat{\mathcal{E}}} \exp W_{ij} \prod_{\nu_i\in\mathcal{V}} \exp \psi_i \left(\deg\left(\nu_i, \hat{\mathcal{E}}\right)\right). \quad (1)$$

The singleton edge potentials are represented by a matrix $W \in \mathbb{R}^{n\times n}$ where W_{ij} is the gain in log-likelihood when edge (i, j) is changed from off to on. The functions $\psi_i : \{1, \ldots, n\} \to \mathbb{R}$ where $j \in \{1, \ldots n\}$ are potentials over the degrees of each node with respect to edges $\hat{\mathcal{E}}$. In other words, the probability of an edge structure depends on local edge weight as well as a prior degree bias. Unfortunately, due to the many dependencies implicated in each degree distribution term ψ_j, the probability model above has large tree-width. Therefore, exact inference and naive MAP estimation procedures (for instance, using the junction tree algorithm) can scale exponentially with $|V|$. However, with clever construction, exact MAP estimation is possible when the degree potentials are concave.

2.1 Encoding as a b-matching

If we make the mild assumption that the ψ_i functions in Eq. 1 are concave, the probability of interest can be maximized by solving a b-matching. By concavity, we mean that the change induced by increasing the input degree must be monotonically non-increasing. This is the standard notion of concavity if ψ_i is made continuous by linearly interpolating between integral degrees. Formally,

$$\begin{aligned}
\delta\psi_i(k) &= \psi_i(k) - \psi_i(k-1), \\
\delta^2\psi_i(k) &= \delta\psi_i(k) - \delta\psi_i(k-1) \\
&= \psi_i(k) - \psi_i(k-1) - \\
&\quad\ (\psi_i(k-1) - \psi_i(k-2)) \leq 0.
\end{aligned}$$

When degree potentials are concave, we can exactly mimic the probability function $\Pr(\hat{\mathcal{E}}|\mathcal{G})$ by building a larger graph with corresponding probability $\Pr(\hat{\mathcal{E}}_b|\mathcal{G}_b)$.

Our construction proceeds as follows. First create a new graph $\mathcal{G}_b$, which contains a copy of the original graph $\mathcal{G}$ as well as additional dummy nodes denoted $\mathcal{D}$. These dummy nodes mimic the role of the soft degree potential functions ψ_i. For each node ν_i in our original set $\mathcal{V}$, we introduce a set of dummy nodes. We add one dummy node for each edge in $\mathcal{E}$ that is adjacent to each ν_i. In other words, for each node ν_i, we add dummy nodes $d_{i,1}, \ldots, d_{i,N_i}$ where $N_i = \deg(\nu_i, \mathcal{E})$ is the size of the neighborhood of node ν_i. Each of the dummy nodes in $d_{i,1}, \ldots, d_{i,N_i}$ is connected to ν_i in the new graph $\mathcal{G}_b$. This construction creates graph $\mathcal{G}_b = \{\mathcal{V}_b, \mathcal{E}_b\}$ defined as follows:

$$\begin{aligned}
\mathcal{D} &= \{d_{1,1}, \ldots, d_{1,N_1}, \ldots, d_{n,1}, \ldots, d_{n,N_n}\}, \\
\mathcal{V}_b &= \mathcal{V} \cup \mathcal{D}, \\
\mathcal{E}_b &= \mathcal{E} \cup \{(\nu_i, d_{i,j})|1 \leq j \leq N_i, 1 \leq i \leq n\}.
\end{aligned}$$

We next specify the weights of the edges in $\mathcal{G}_b$. The weight of each edge (i, j) is copied from $\mathcal{E}$ to its original potential W_{ij}. We set the edge weights between the original nodes and dummy nodes according to the following formula. The potential between ν_i and each dummy node $d_{i,j}$ is

$$w(\nu_i, d_{i,j}) = \psi_i(j-1) - \psi_i(j). \quad (2)$$

While the ψ functions have outputs for $\psi(0)$, there are no dummy nodes labeled $d_{i,0}$ associated with that setting ($\psi(0)$ is only used when defining the weight of $d_{i,1}$). By construction, the weights $w(\nu_i, d_{i,j})$ are monotonically non-decreasing with respect to the index j due to the concavity of the ψ functions. This characteristic leads to the guaranteed correctness of our method.

$$\begin{aligned}
\psi_i(j) - \psi_i(j-1) &\leq \psi_i(j-1) - \psi_i(j-2) \\
-w(\nu_i, d_{i,j}) &\leq -w(\nu_i, d_{i,j-1}) \\
w(\nu_i, d_{i,j}) &\geq w(\nu_i, d_{i,j-1}). \quad (3)
\end{aligned}$$

We emulate the probability $\Pr(\hat{\mathcal{E}}|\mathcal{G})$ (Eq. 1), which is over edges in $\mathcal{G}$, with a probability $\Pr(\hat{\mathcal{E}}_b|\mathcal{G}_b)$, which is over edges of $\mathcal{G}_b$. We set the degree constraints such that each (original) node ν_i must have exactly N_i neighbors (including any connected dummy nodes to which it might connect). Dummy nodes have no degree constraints. The proposed approach recovers the most likely subgraph $\hat{\mathcal{E}}_b =$

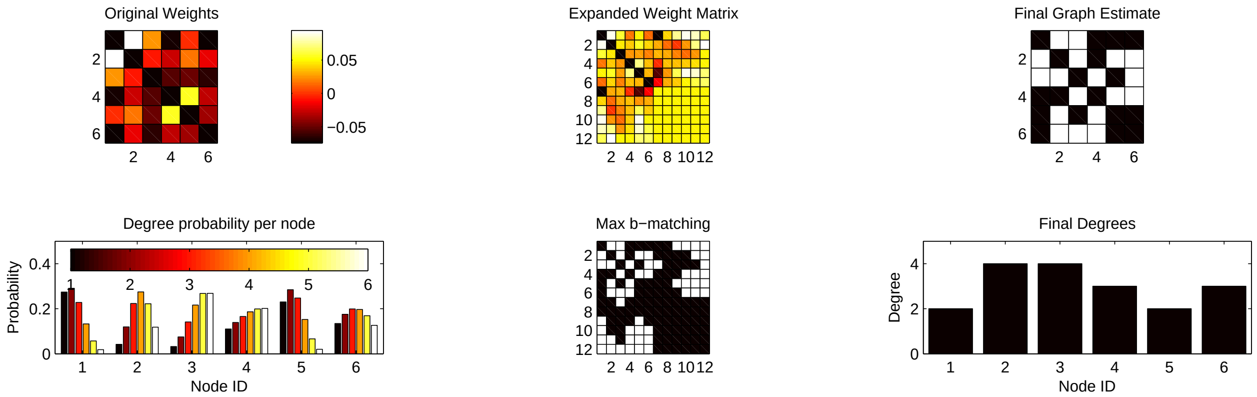

Figure 1. Example of mapping a degree dependent problem to a hard-constrained b-matching. Left: Original weight matrix and row/column degree distributions. Upper Middle: Weight matrix of expanded graph, whose solution is now constrained to have exactly 6 neighbors per node. Lower Middle: Resulting b-matching, whose upper left quadrant is the final output. Right: MAP solution and final node degrees.

$$\left[\begin{array}{cccccc}
W_{1,1} & W_{1,2} & \cdots & W_{1,n} & \psi_1(0)-\psi_1(1) & \cdots & \psi_1(n-1)-\psi_1(n) \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
W_{n,1} & W_{n,2} & \cdots & W_{n,n} & \psi_n(0)-\psi_n(1) & \cdots & \psi_n(n-1)-\psi_2(n) \\
\psi_1(0)-\psi_1(1) & \psi_2(0)-\psi_2(1) & \cdots & \psi_n(0)-\psi_n(1) & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\psi_1(n-1)-\psi_1(n) & \psi_2(n-1)-\psi_2(n) & \cdots & \psi_n(n-1)-\psi_n(n) & 0 & \cdots & 0
\end{array}\right]$$

Figure 2. The new weight matrix constructed by the procedure in Section 2. The upper left quadrant is the original weight matrix, and the extra rows and columns are the weights for dummy edges.

$\arg\max_{\hat{\mathcal{E}}_b} \Pr(\hat{\mathcal{E}}_b|\mathcal{G}_b)$ by solving the following b-matching problem:

$$\hat{\mathcal{E}}_b = \arg\max_{\hat{\mathcal{E}}_b \subseteq \mathcal{E}_b} \sum_{(\nu_i, d_{i,j}) \in \hat{\mathcal{E}}_b} w(\nu_i, d_{i,j}) + \sum_{(i,j) \in \hat{\mathcal{E}}_b} W_{ij}$$
$$\text{subject to} \quad \deg(\nu_i, \hat{\mathcal{E}}_b) = N_i \text{ for } \nu_i \in \mathcal{V}. \quad (4)$$

This construction can be conceptualized in the following way: we are free to choose any graph structure in the original graph, but pay a penalty based on node degrees due to selecting dummy edges maximally. The following theorem proves that this penalty is equivalent to that created by the degree priors.

Theorem 1. *The total edge weight of b-matchings $\hat{\mathcal{E}}_b = \arg\max_{\hat{\mathcal{E}}_b} \log \Pr(\hat{\mathcal{E}}_b|\mathcal{G}_b)$ from graph $\mathcal{G}_b$ differs from $\log \Pr(\hat{\mathcal{E}}_b \cap \mathcal{E}|\mathcal{G})$ by a fixed additive constant.*

Proof. Consider the edges $\hat{\mathcal{E}}_b \cap \mathcal{E}$. These are the estimated connectivity $\hat{\mathcal{E}}$ after we remove dummy edges from $\hat{\mathcal{E}}_b$. Since we set the weight of the original edges to the W_{ij} potentials, the total weight of these edges is exactly the first term in (1), the local edge weights.

What remains is to confirm that the ψ degree potentials agree with the weights of the remaining edges $\hat{\mathcal{E}}_b \setminus (\hat{\mathcal{E}}_b \cap \mathcal{E})$ between original nodes and dummy nodes. Recall that our degree constraints require each node in $\mathcal{G}_b$ to have degree N_i. By construction, each ν_i has $2N_i$ available edges from which to choose: N_i edges from the original graph and N_i edges to dummy nodes. Moreover, if ν_i selects k original edges, it must maximally select $N_i - k$ dummy edges. Since the dummy edges are constructed such that their weights are non-decreasing, the maximum $N_i - k$ dummy edges are to the last $N_i - k$ dummy nodes, or dummy nodes $d_{i,k+1}$ through d_{i,N_i}. Thus, we must verify the following:

$$\sum_{j=k+1}^{N_i} w(\nu_i, d_{i,j}) - \sum_{j=k'+1}^{N_i} w(\nu_i, d_{i,j}) \overset{?}{=} \psi_i(k) - \psi_i(k').$$

Terms in the summations cancel out to show this equivalence. After substituting the definition of $w(\nu_i, d_{i,j})$, the desired equality is revealed.

$$\begin{aligned}\sum_{j=k+1}^{N_i} (\psi_i(j-1) - \psi_i(j)) - \sum_{j=k'+1}^{N_i} (\psi_i(j-1) - \psi_i(j)) \\ = \sum_{j=k}^{N_i} \psi_i(j) - \sum_{j=k+1}^{N_i} \psi_i(j) - \sum_{j=k'}^{N_i} \psi_i(j) + \sum_{j=k'+1}^{N_i} \psi_i(j) \\ = \psi_i(k) - \psi_i(k')\end{aligned}$$

This means the log-probability and the weight of the new graph change the same amount as we try different subgraphs of $\mathcal{G}$. Hence, for any b-matching $\hat{\mathcal{E}}_b$, the quantities $\log \Pr(\hat{\mathcal{E}}_b \cap \mathcal{E}|\mathcal{G})$ and $\max_{\hat{\mathcal{E}}_b \setminus \mathcal{E}} \log \Pr(\hat{\mathcal{E}}_b|\mathcal{G}_b)$ differ only by a constant. □

2.2 Computational cost and generalized methods

Since the dummy nodes have no degree constraints, we only need to instantiate $\max_i(N_i)$ dummy nodes and reuse them for each node ν_i. The process described in this section is illustrated in Figures 2 and 1. This results in at most a twofold increase of total nodes in the constructed graph (i.e., $|V_b| \leq 2|V|$). In practice, we can find the maximum weight b-matching to maximize $\Pr(\hat{\mathcal{E}}_b|\mathcal{G}_b)$ using classical maximum flow algorithms [2], which require $\mathcal{O}(|\mathcal{V}_b||\mathcal{E}_b|)$ computation time. However, in the special case of bipartite graphs, we can use belief propagation [1, 5, 16], which yields not only a rather large constant factor speedup, but has been theoretically proven to find the solution in $(|\mathcal{V}_b|^2)$ or $(|\mathcal{E}_b|)$ time under certain mild assumptions [15]. Furthermore, the algorithm can be shown to obtain exact solutions in the unipartite case when linear programming integrality can be established [16, 6].

The class of log-concave degree priors generalizes many maximum weight constrained-subgraph problems. These include simple thresholding of the weight matrix, which is implemented by placing an exponential distribution on the degree; setting the degree prior to $\psi_i(k) = -\theta k$ causes the maximum to have edges on when W_{ij} is greater than threshold θ. We can mimic b-matching by setting the degree priors to be delta-functions at degree b. We can mimic bd-matching, which enforces lower and upper bounds on the degrees, by setting the degree priors to be uniform between the bounds and to have zero probability elsewhere. We can mimic k-nearest neighbors by duplicating the nodes of the graph to form a bipartite graph, where edges between nodes in the original graph are represented by edges between bipartitions, and by setting the degrees of one bipartition to exactly k while having no constraints on the other bipartition.

3 Experiments

We apply the MAP estimation algorithm as a post-processing step in a graph prediction problem. Consider the task of predicting a graph defined by the preferences of users to items in a slight variation of the standard collaborative filtering setting. We define a preference graph as a bipartite graph between a set of users $U = \{u_1, \ldots, u_n\}$ and a set of items $V = \{v_1, \ldots, v_m\}$ that the users have rated with binary recommendations. We assume a rating matrix $Y = \{0, 1\}^{n \times m}$ representing the preferences of users (rows) for items (columns). The rating matrix Y is equivalent to the adjacency matrix of the preference graph, and $Y_{ij} = 1$ indicates that user i approves of item j while $Y_{ij} = 0$ indicates that the user disapproves. The training data is a set of user-item pairs and whether an edge is present between their nodes in the preference graph. The testing data is another set of user-item pairs, and the task is to predict which of the testing pairs will have a preference edge present.

First, we provide motivation for using degree priors in post-processing. The degrees of nodes in the predicted graph represent the number of items liked by a user or the number of users that like an item. Under certain assumptions, we can prove that the rate of liking or being liked will concentrate around its empirical estimate, and the deviation probability between training and testing rates is bounded by a log-concave upper bound. Therefore, we will use the deviation bound as a degree prior to post-process predictions output by a state of the art inference method. This, in effect, forces our predictions to obey the bounds.

3.1 Concentration bound

We assume that users U and items V are drawn *iid* from arbitrary population distributions $\mathbb{D}_u$ and $\mathbb{D}_v$. We also assume that the probability of an edge between any nodes u_i and v_j is determined by a function that maps the features of the nodes to a valid Bernoulli probability.

$$\Pr(Y_{ij} = 1|u_i, v_j) = f(u_i, v_j) \in [0, 1]. \qquad (5)$$

These assumptions yield a natural dependency structure for rating probabilities. The joint probability of users, items and ratings is defined as follows:

$$\begin{aligned}\Pr(Y, U, V|\mathbb{D}_u, \mathbb{D}_v) \propto \\ \prod_{ij} p(Y_{ij}|u_i, v_j) \prod_i p(u_i|\mathbb{D}_u) \prod_j p(v_j|\mathbb{D}_v).\end{aligned} \qquad (6)$$

The structure of this generative model implies dependencies between the unobserved ratings and even dependencies between the users and movies. This is because the query rating variables and all user and item variables are latent.

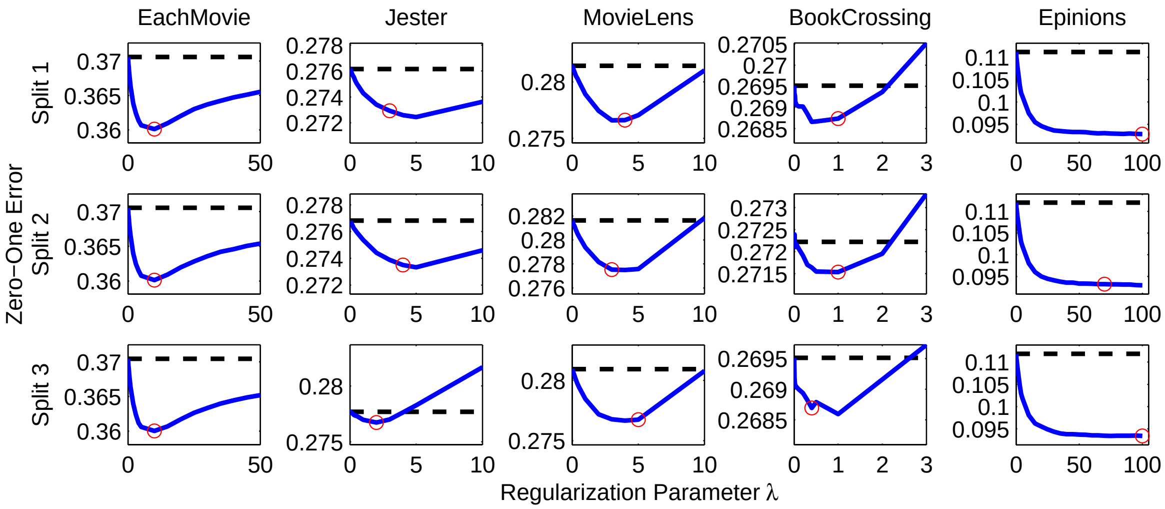

Figure 3. Testing errors of MAP solution across different data sets and random splits. The x-axis of each plot represents the scaling parameter λ and the y-axis represents the error rate. The solid blue line is the MAP solution with degree priors, the dotted black line is the logistic-fMMMF baseline. The red circle marks the setting of λ that performed best on the cross-validation set. See Table 1 for the numerical scores.

Due to the independent sampling procedure on users and items, this is known as a hierarchical model [3] and induces a coupling, or interdependence, between the test predictions that are to be estimated by the algorithm. Since the rating variables exist in a lattice of common parents, this dependency structure and the hierarchical model are difficult to handle in a Bayesian setting unless strong parametric assumptions are imposed. Instead, we next derive a bound that captures the interdependence of the structured output variables Y without parametric assumptions.

We assume that both the training and testing user-item sets are completely randomly revealed from a set of volunteered ratings, which allows proof of an upper bound for the probability that the empirical edge rate of a particular node deviates between training and testing data. In other words, we estimate the probability that an empirical row or column average in the adjacency matrix deviates from its true mean. Without loss of generality, let the training ratings for user i be at indices $\{1, \ldots, c_{\rm tr}\}$ and the testing ratings be at indices $\{c_{\rm tr}+1, \ldots, c_{\rm tr}+c_{\rm te}\}$ such that the training and testing sets are respectively of size $c_{\rm tr}$ and $c_{\rm te}$.[1] Let $\bar{Y}_i = [Y_{i,1}, \ldots, Y_{i,c_{\rm tr}+c_{\rm te}}]$ represent the row of ratings by user i. Let function $\Delta(\bar{Y}_i)$ represent the difference between the training and query averages. The following theorem bounds the difference between training and testing rating averages:

$$\Delta(Y_{i,1}, \ldots, Y_{i,c_{\rm tr}+c_{\rm te}}) = \frac{1}{c_{\rm tr}} \sum_{j=1}^{c_{\rm tr}} Y_{ij} - \frac{1}{c_{\rm te}} \sum_{j=c_{\rm tr}+1}^{c_{\rm tr}+c_{\rm te}} Y_{ij},$$

which will obey the following theorem.

Theorem 2. *Given that users $U = \{u_1, \ldots, u_n\}$ and rated items $V = \{v_1, \ldots, v_n\}$ are drawn* iid *from arbitrary distributions $\mathbb{D}_u$ and $\mathbb{D}_v$ and that the probability of positive rating by a user for an item is determined by a function $f(u_i, v_j) \mapsto [0,1]$, the average of query ratings by each user is concentrated around the average of his or her training ratings. Formally,*

$$\begin{aligned} \Pr\left(\Delta(\bar{Y}_i) \geq \epsilon\right) &\leq 2\exp\left(-\frac{\epsilon^2 c_{\rm tr} c_{\rm te}}{2(c_{\rm tr}+c_{\rm te})}\right), \quad (7) \\ \Pr\left(\Delta(\bar{Y}_i) \leq -\epsilon\right) &\leq 2\exp\left(-\frac{\epsilon^2 c_{\rm tr} c_{\rm te}}{2(c_{\rm tr}+c_{\rm te})}\right). \end{aligned}$$

The proof of Theorem 2 is deferred to Appendix A.

Using a standard learning method, we learn the estimates of each edge. However, predicting the most likely setting of each edge independently is equivalent to using a uniform prior over the rating averages. However, a uniform prior violates the bound at a large enough deviation from the training averages. Specifically, this occurs for users or items with a large number of training and testing examples. Thus, it may be advantageous to use a prior that obeys the bound.

[1] We omit a subscript for the training and testing counts $c_{\rm tr}$ and $c_{\rm te}$ for notational clarity only. Since these counts vary for different nodes, precise notation would involve terms such as $c_i^{\rm tr}, c_i^{\rm te}$.

Since the bound decays quadratically in the exponent, priors that will never violate the bound must decay at a faster rate. These exclude uniform and Laplace distributions and include Gaussian, sub-Gaussian and delta distributions. We propose simply using the normalized bound as a prior.

3.2 Edge weights

To learn reasonable values for the independent edge weights, we employ Fast Max-Margin Matrix Factorization (fMMMF) [14] using a logistic loss function, which has a natural probabilistic interpretation [13]. In the binary-ratings setting, the gradient optimization for logistic fMMMF, which uses a logistic loss as a differential approximation of hinge-loss, can be interpreted as maximizing the conditional likelihood of a generative model that is very similar to one discussed above. The objective is [2]

$$\min_{U,V} J(U,V) = \frac{1}{2}(||U||^2_{\text{Fro}} + ||V||^2_{\text{Fro}}) + C\sum_{ij} \log\left(1 + e^{-Y^{\pm}_{ij}(u_i^\top v_j - \theta_i)}\right). \tag{8}$$

The probability function for positive ratings is the logistic function, which yields the exact loss term above.

$$\Pr(Y_{ij}|u_i, v_j, \theta_i) = f(u_i, v_j) = \frac{1}{1 + e^{-(u_i^\top v_j - \theta_i)}}$$

Minimization of squared Frobenius norm corresponds to placing zero-mean, spherical Gaussian priors on the u_i and u_i vectors, $\Pr(u_i) \propto \exp(-\frac{1}{C}||u_i||^2)$ and $\Pr(v_j) \propto \exp(-\frac{1}{C}||v_j||^2)$. This yields the interpretation of fMMMF as MAP estimation [13]:

$$\max_{U,V,\Theta} \prod_{ij} P(Y_{ij}|u_i, v_j, \theta_i) \prod_i \Pr(u_i) \prod_j \Pr(v_j).$$

Once we find the MAP U and V matrices using fMMMF, we use the logistic probabilities to set the singleton functions over edges (i.e., edge weights). Specifically, the weight of an edge is the change in log-likelihood caused by switching the edge from inactive to active, $W_{ij} = u_i^\top v_j - \theta_i$.

3.3 Results

Our experiments tested five data sets. Four are standard collaborative filtering datasets that we thresholded at reasonable levels. The last is trust/distrust data gathered from Epinions.com which represents whether users trust other users' opinions. The EachMovie data set contains 2,811,983 integer ratings by 72,916 users for 1,628 movies ranging from 1 to 6, which we threshold at 4 or greater to represent a positive rating. The portion of the Jester data set [4] we used contains 1,810,455 ratings by 24,983 users for 100 jokes ranging from -10 to 10, which we threshold at 0 or greater. The MovieLens-Million data set contains 1,000,209 integer ratings by 6,040 users for 3,952 movies ranging from 1 to 5, which we threshold at 4 or greater. The Book Crossing data set [19] contains 433,669 explicit integer ratings[3] by 77,805 users for 185,854 books ranging from 1 to 10, which we threshold at 7 or greater. Lastly, the Epinions data set [9] contains 841,372 trust/distrust ratings by 84,601 users for 95,318 authors.

Each data set is split randomly three times into half training and half testing ratings. We randomly set aside $1/5$ of the training set for cross-validation, and train logistic fMMMF on the remainder using a range of regularization parameters. The output of fMMMF serves as both our baseline as well as the weight matrix of our algorithm. We set the "degree" distribution for each row/column to be proportional to the deviation bound from Theorem 2. Specifically, we use the following formula to set the degree potential ψ_i:

$$\psi_i(k) = -\lambda \frac{\left(\frac{1}{c_{\text{tr}}}\sum_{j=1}^{c_{\text{tr}}} Y_{ij} - k/c_{\text{te}}\right)^2 c_{\text{tr}} c_{\text{te}}}{2(c_{\text{tr}} + c_{\text{te}})} \tag{9}$$

We introduce a regularization parameter λ that scales the potentials. When λ is zero, the degree prior becomes uniform and the MAP solution is to threshold the weight matrix at 0 (the default fMMMF predictions). At greater values, we move from a uniform degree prior (default rounding) toward strict b-matching, following the shape of the concentration bound at intermediary settings. We explore increasing values of λ starting at 0 until either the priors are too restrictive and we observe overfitting or until the value of λ is so great that we are solving a simple b-matching with degrees locked to an integer value instead of a distribution of integers. Increasing λ thereafter will not change the result. We cross-validate at this stage by including the testing and held-out cross-validation ratings in the query set of ratings.

The running time of the post-processing procedure is short compared to the time spent learning edge weights via fMMMF. This is due to the fast belief propagation matching code and the sparsity of the graphs. Each graph estimation takes a few minutes (no more than five), while the gradient fMMMF takes hours on these large-scale data sets.

We compare the zero-one error of prediction on the data. In particular, we are interested in comparing the fMMMF output that performed best on cross-validation data to the

[2] Here $Y^{\pm}_{ij}$ represents the signed $\{-1, +1\}$ representation of the binary rating, whereas previously, we use the $\{0, 1\}$ representation.

[3] The Book Crossing data set contains many more "implicit" recommendations, which occur when users purchase books but do not explicitly rate them. Presumably, these indicate positive opinions of the books; however, it is difficult to define a negative implicit rating, so we only experiment on the explicit ratings.

Table 1. Average zero-one error rates and standard deviations of best MAP with degree priors and fMMMF chosen via cross-validation. Average taken over three random splits of the data sets into testing and training data. Degree priors improve accuracy on all data sets, but statistically significant improvements according to a two-sample t-test with a rejection level of 0.01 are bold.

Data set	fMMMF	Degree
EachMovie	0.3150 ± 0.0002	**0.2976 ± 0.0001**
Jester	0.2769 ± 0.0008	0.2744 ± 0.0021
MovieLens	0.2813 ± 0.0004	**0.2770 ± 0.0005**
BookCrossing	0.2704 ± 0.0016	0.2697 ± 0.0016
Epinions	0.1117 ± 0.0005	**0.0932 ± 0.0003**

MAP solution of the same output with additional degree priors. The results indicate that adding degree priors reduces testing error on all splits of five data sets. The error rates are represented graphically in Fig. 3 and numerically in Table 1. With higher λ values, the priors pull the prediction averages closer to the training averages, which causes overfitting on all but the Epinions data set. Interestingly, even b-matching the Epinions data set improves the prediction accuracy over fMMMF. This suggests that the way users decide whether they trust other users is determined by a process that is strongly concentrated. While choosing the bound as a prior and the sampling assumptions made in this article may be further refined in future work, it is important to note that enforcing degree distribution properties on the estimated graph consistently helps improve the performance of a state of the art factorization approach.

4 Discussion

We have provided a method to find the most likely graph from a distribution that uses edge weight information as well as degree distributions for each node. The exact MAP estimate is computed in polynomial time by showing that the problem is equivalent to b-matching or maximum weight degree constrained subgraph problem. These can be efficiently and exactly implemented using maximum flow as well as faster belief propagation methods. Our method generalizes b-matching, bd-matching, simple thresholding, and k-nearest neighbors which can all be viewed as graph structure estimation with different degree distributions. Various methods that use either these simple degree distributions or no degree information at all may benefit from generalizing the degree information to allow for uncertainty. The main limitation of the approach is that the degree distributions that can be modeled in this way must be log-concave, thus exact inference with more general degree distributions is an open problem.

References

[1] M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. In *Proc. of the IEEE International Symposium on Information Theory*, 2005.

[2] C. Fremuth-Paeger and D. Jungnickel. Balanced network flows. i. a unifying framework for design and analysis of matching algorithms. *Networks*, 33(1), 1999.

[3] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC, July 2003.

[4] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Inf. Retr.*, 4(2), 2001.

[5] B. Huang and T. Jebara. Loopy belief propagation for bipartite maximum weight b-matching. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, volume 2 of JMLR: W&CP, March 2007.

[6] T. Jebara. Map estimation, message passing, and perfect graphs. In *Uncertainty in Artificial Intelligence*, 2009.

[7] T. Jebara, J. Wang, and S.F. Chang. Graph construction and b-matching for semi-supervised learning. In *International Conference on Machine Learning*, 2009.

[8] P. M. Kim, L. J. Lu, Y. Xia, and M. B. Gerstein. Relating three-dimensional structures to protein networks provides evolutionary insights. *Science*, 314(5807):1938–41, 2006 Dec 22.

[9] P. Massa and P. Avesani. Controversial users demand local trust metrics: An experimental study on epinions.com community. In Manuela M. Veloso and Subbarao Kambhampati, editors, *AAAI*. MIT Press, 2005.

[10] C. McDiarmid. On the method of bounded differences. *Surveys in Combinatorics*, 1989.

[11] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized online matching. *J. ACM*, 54(5), 2007.

[12] Q. Morris and B. Frey. Denoising and untangling graphs using degree priors. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

[13] J. Rennie. *Extracting information from informal communication*. PhD thesis, MIT, 2007.

[14] J. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005.

[15] J. Salez and D. Shah. Optimality of belief propagation for random assignment problem. In Claire Mathieu, editor, *SODA*, pages 187–196. SIAM, 2009.

[16] S. Sanghavi, D. Malioutov, and A. Willsky. Linear programming analysis of loopy belief propagation for weighted matching. In *Advances in Neural Information Processing Systems 20*. MIT Press, 2008.

[17] B. Shaw and T. Jebara. Minimum volume embedding. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, volume 2 of JMLR: W&CP, March 2007.

[18] B. Shaw and T. Jebara. Structure preserving embedding. In *International Conference on Machine Learning*, 2009.

[19] C.-N. Ziegler, S. McNee, J. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In Allan Ellis and Tatsuya Hagino, editors, *WWW*. ACM, 2005.

Appendix A

Proof of Theorem 2. McDiarmid's Inequality bounds the deviation probability of a function over independent (but not necessarily identical) random variables from its expectation in terms of its Lipschitz constants [10], which are the maximum change in the function value induced by changing any input variable. The Lipschitz constants for function Δ are $\ell_j = 1/c_{\mathrm{tr}}$ for $1 \leq j \leq c_{\mathrm{tr}}$, and $\ell_j = 1/c_{\mathrm{te}}$ otherwise. Although the rating random variables are not identically distributed, they are independently sampled, so we can apply McDiarmid's Inequality (and simplify) to obtain

$$\Pr\left(\Delta(\bar{Y_i}) - \mathrm{E}[\Delta] \geq t\right) \quad \leq \quad \exp\left(-\frac{2t^2 c_{\mathrm{tr}} c_{\mathrm{te}}}{c_{\mathrm{tr}} + c_{\mathrm{te}}}\right) (10)$$

The left-hand side quantity inside the probability contains $\mathrm{E}[\Delta]$, which should be close to zero, but not exactly zero (if it were zero, Eq. 10 would be the bound). Since our model defines the probability of Y_{ij} as a function of u_i and v_j, the expectation is

$$\begin{aligned} E\left[\Delta(\bar{Y_i})\right] &= E\left[\frac{1}{c_{\mathrm{tr}}}\sum_{j=1}^{c_{\mathrm{tr}}} Y_{ij} - \frac{1}{c_{\mathrm{te}}}\sum_{j=c_{\mathrm{tr}}+1}^{c_{\mathrm{tr}}+c_{\mathrm{te}}} Y_{ij}\right] \\ &= \frac{1}{c_{\mathrm{tr}}}\sum_{j=1}^{c_{\mathrm{tr}}} f(u_i, v_j) - \frac{1}{c_{\mathrm{te}}}\sum_{j=c_{\mathrm{tr}}+1}^{c_{\mathrm{tr}}+c_{\mathrm{te}}} f(u_i, v_j) \\ &\overset{\mathrm{def}}{=} g_i(V) \end{aligned}$$

We define the quantity above as a function over the items $V = \{v_1, \ldots, v_{c_{\mathrm{tr}}+c_{\mathrm{te}}}\}$, which we will refer to as $g_i(V)$ for brevity. Because this analysis is of one user's ratings, we can treat the user input u_i to $f(u_i, v_j)$ as a constant. Since the range of the probability function $f(u_i, v_i)$ is $[0, 1]$, the Lipschitz constants for $g_i(V)$ are $\ell_j = 1/c_{\mathrm{tr}}$ for $1 \leq j \leq c_{\mathrm{tr}}$, and $\ell_j = 1/c_{\mathrm{te}}$ otherwise. We apply McDiarmid's Inequality again.

$$\Pr\left(g_i(V) - \mathrm{E}[g_i(V)] \geq \tau\right) \leq \exp\left(-\frac{2\tau^2 c_{\mathrm{tr}} c_{\mathrm{te}}}{c_{\mathrm{tr}} + c_{\mathrm{te}}}\right).$$

The expectation of $g_i(V)$ can be written as the integral

$$\mathrm{E}[g_i(V)] = \int \Pr(v_1, \ldots, v_{c_{\mathrm{tr}}+c_{\mathrm{te}}}) g_i(V) dV.$$

Since the v's are *iid*, the integral decomposes into

$$\begin{aligned} \mathrm{E}[g_i(V)] &= \frac{1}{c_{\mathrm{tr}}}\sum_{j=1}^{c_{\mathrm{tr}}} \int \Pr(v_j) f(u_i, v_j) dv_j \\ &\quad -\frac{1}{c_{\mathrm{te}}}\sum_{j=c_{\mathrm{tr}}+1}^{c_{\mathrm{tr}}+c_{\mathrm{te}}} \int \Pr(v_j) f(u_i, v_j) dv_j. \end{aligned}$$

Since each $\Pr(v_j) = \Pr(v)$ for all j, by a change of variables all integrals above are identical. The expected value $\mathrm{E}[g_i(V)]$ is therefore zero . This leaves a bound on the value of $g_i(V)$.

$$\Pr\left(g_i(V) \geq \tau\right) \exp\left(-\frac{2\tau^2 c_{\mathrm{tr}} c_{\mathrm{te}}}{c_{\mathrm{tr}} + c_{\mathrm{te}}}\right)$$

To combine the bounds, we define a quantity to represent the probability of each deviation. First, let the probability of $g_i(V)$ exceeding some constant τ be $\frac{\delta}{2}$.

$$\frac{\delta}{2} = \exp\left(-\frac{2\tau^2 c_{\mathrm{tr}} c_{\mathrm{te}}}{c_{\mathrm{tr}} + c_{\mathrm{te}}}\right)$$

Second, let the probability of $\Delta(\bar{Y_i})$ exceeding its expectation by more than a constant t also be $\frac{\delta}{2}$,

$$\frac{\delta}{2} = \exp\left(-\frac{2t^2 c_{\mathrm{tr}} c_{\mathrm{te}}}{c_{\mathrm{tr}} + c_{\mathrm{te}}}\right).$$

We can write both t and τ in terms of δ:

$$t = \tau = \sqrt{\frac{c_{\mathrm{tr}} + c_{\mathrm{te}}}{2 c_{\mathrm{tr}} c_{\mathrm{te}} \log \frac{2}{\delta}}}.$$

Define ϵ as the concatenation of deviations t and τ,

$$\epsilon = t + \tau = 2\sqrt{\frac{c_{\mathrm{tr}} + c_{\mathrm{te}}}{2 c_{\mathrm{tr}} c_{\mathrm{te}} \log \frac{2}{\delta}}}.$$

By construction, the total deviation ϵ occurs with probability greater than δ. Solving for δ provides the final bound in Eq. 7. The bound in the other direction follows easily since McDiarmid's Inequality is also symmetric. □

Although the above analysis refers only to the ratings of the user, the generative model we describe is symmetric between users and items. Similar analysis therefore applies directly to item ratings as well.

Corollary 1. *Under the same assumptions as Theorem 2, the average of query ratings for each item is concentrated around the average of its training ratings.*

Additionally, even though Theorem 2 specifically concerns preference graphs, it can be easily extended to show the concentration of edge connectivity in general unipartite and bipartite graphs as follows.

Corollary 2. *The concentration bound in Theorem 2 applies to general graphs; assuming that edges and non-edges are revealed randomly, nodes are generated* iid *from some distribution and the probability of an edge is determined by a function of its vertices, the average connectivity of unobserved (testing) node-pairs is concentrated around the average connectivity of observable (training) node-pairs. The probability of deviation is bounded by the same formula as in Theorem 2.*